Q1. Search Algorithms Potpourri

(a) We will investigate various search algorithms for the following graph. Edges are labeled with their costs, and heuristic values $h$ for states are labeled next to the states. $S$ is the start state, and $G$ is the goal state. In all search algorithms, assume ties are broken in alphabetical order.

(i) Select all boxes that describe the given heuristic values.

- [ ] admissible
- [ ] consistent
- [x] Neither

(ii) Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run A* graph search with the heuristic values provided.

<table>
<thead>
<tr>
<th>Index</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Not Expanded</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>●</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(iii) Assuming we run A* graph search with the heuristic values provided, what path is returned?

- ○ $S \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow G$
- ○ $S \rightarrow A \rightarrow C \rightarrow G$
- ○ $S \rightarrow A \rightarrow C \rightarrow D \rightarrow G$
- ● $S \rightarrow B \rightarrow C \rightarrow G$
- ○ $S \rightarrow A \rightarrow C \rightarrow D \rightarrow G$
- ○ $S \rightarrow A \rightarrow B \rightarrow C \rightarrow G$

(iv) Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run greedy graph search with the heuristic values provided.
**What path is returned by greedy graph search?**

- $S \rightarrow A \rightarrow B \rightarrow C \rightarrow D \rightarrow G$
- $S \rightarrow A \rightarrow C \rightarrow G$
- $S \rightarrow A \rightarrow C \rightarrow D \rightarrow G$
- None of the above

**Consider a complete graph, $K_n$, the undirected graph with $n$ vertices where all $n$ vertices are connected (there is an edge between every pair of vertices), resulting in $\binom{n}{2}$ edges. Please select the maximum possible depth of the resulting tree when the following graph search algorithms are run (assume any possible start and goal vertices).**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$1$</th>
<th>$\left\lceil \frac{n}{2} \right\rceil$</th>
<th>$n - 1$</th>
<th>$\binom{n}{2}$</th>
<th>None of the above</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFS</td>
<td>$\bullet$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\circ$</td>
</tr>
<tr>
<td>DFS</td>
<td>$\circ$</td>
<td>$\circ$</td>
<td>$\bullet$</td>
<td>$\circ$</td>
<td>$\circ$</td>
</tr>
</tbody>
</table>

**Given two admissible heuristics $h_A$ and $h_B$.**

**Which of the following are guaranteed to also be admissible heuristics?**

- $h_A + h_B$
- $\frac{1}{2}(h_A)$
- $\frac{1}{2}(h_B)$
- $\frac{1}{2}(h_A + h_B)$
- $h_A \cdot h_B$
- $\max(h_A, h_B)$
- $\min(h_A, h_B)$

**Consider performing A* tree search. Which is generally best to use if we want to expand the fewest number of nodes?**

- $h_A + h_B$
- $\frac{1}{2}(h_A)$
- $\frac{1}{2}(h_B)$
- $\frac{1}{2}(h_A + h_B)$
- $h_A \cdot h_B$
- $\max(h_A, h_B)$
- $\min(h_A, h_B)$

**Consider performing tree search for some search graph. Let $\text{depth}(n)$ be the depth of search node $n$ and $\text{cost}(n)$ be the total cost from the start state to node $n$. Let $G_d$ be a goal node with minimum depth, and $G_c$ be a goal node with minimum total cost. Assume edge costs $> 0$.**

**For iterative deepening (where we repeatedly run DFS and increase the maximum depth allowed by 1), mark all conditions that are guaranteed to be true for every node $n$ that could be expanded during the search, or mark ”None of the above” if none of the conditions are guaranteed.**

<table>
<thead>
<tr>
<th>Condition</th>
<th>Mark</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{cost}(n) \leq \text{cost}(G_c)$</td>
<td>$\Box$</td>
</tr>
<tr>
<td>$\text{cost}(n) \leq \text{cost}(G_d)$</td>
<td>$\Box$</td>
</tr>
<tr>
<td>$\text{depth}(n) \leq \text{depth}(G_c)$</td>
<td>$\Box$</td>
</tr>
<tr>
<td>$\text{depth}(n) \leq \text{depth}(G_d)$</td>
<td>$\Box$</td>
</tr>
<tr>
<td>None of the above</td>
<td>$\Box$</td>
</tr>
</tbody>
</table>

When running iterative deepening we will explore all nodes of depth $k$, before we explore any nodes of depth $k + 1$. As a result will never explore any nodes that have depth greater than $G_d$ because we would stop exploring once we reached $G_d$. This also means we would never explore any nodes with depth greater than $G_c$ because $G_c$ has to have depth greater than or equal to the minimum depth goal, $G_d$

**What is necessarily true regarding iterative deepening on any search tree?**

- Complete as opposed to DFS tree search
- Strictly faster than DFS tree search
- Strictly faster than BFS tree search
- More memory efficient than BFS tree search
- A type of stochastic local search
- None of the above